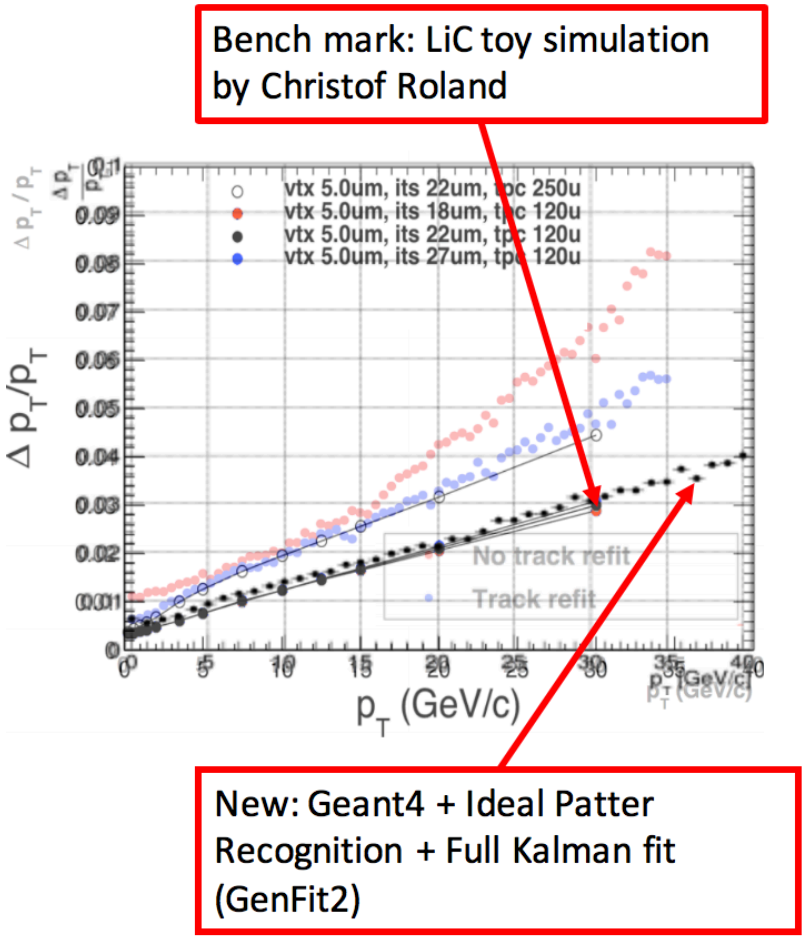
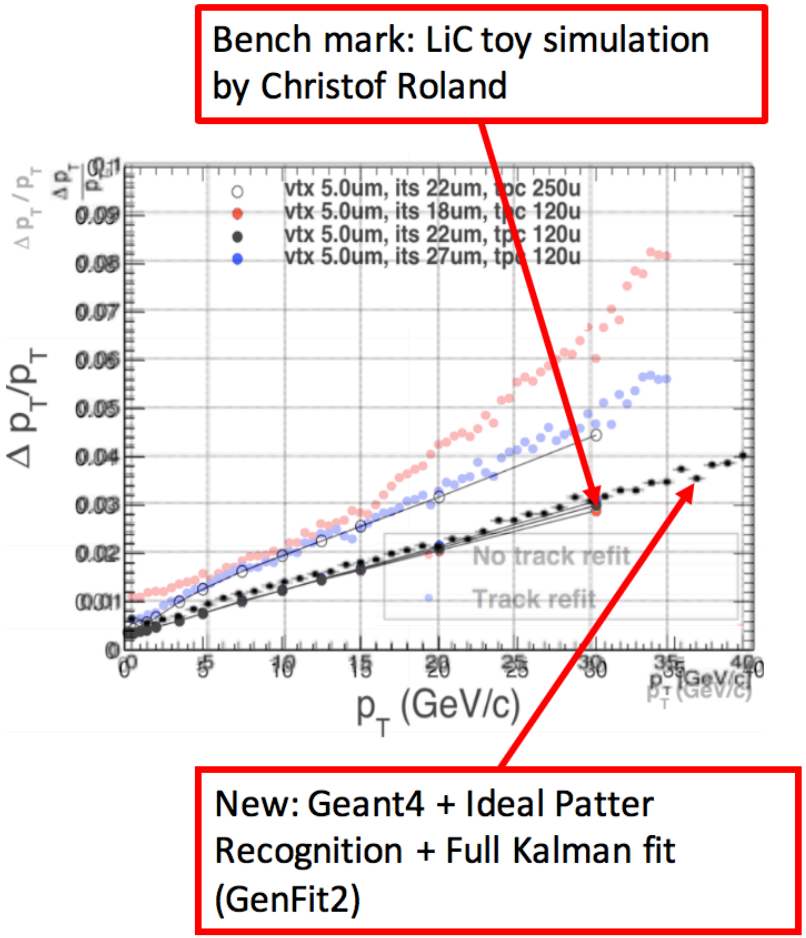
sPhenix Tracking Refurbishment

The Challenge



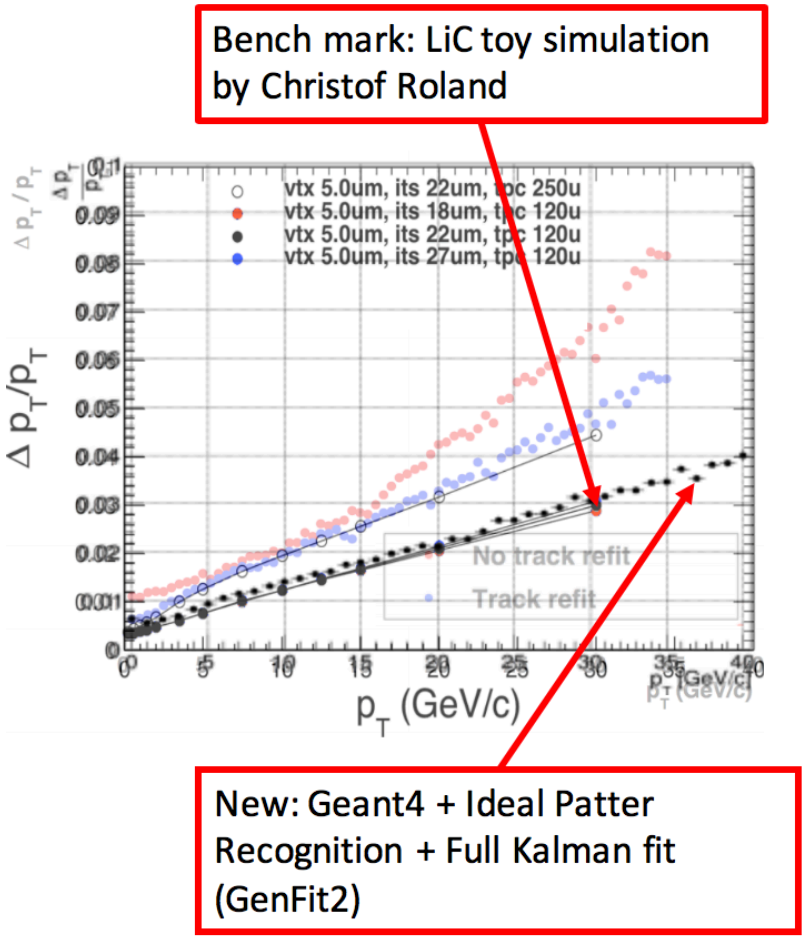
- The current tracking algorithm doesn't seam to reach the full potential of the sPhenix tracker
  - Slow
  - Non ideal parameter estimation
- Need to document the HF reconstruction capabilities of the sPhenix detector including the MAPS telescope for the MVTX proposal in ~Summer
  - Not a lot of time

The Upside



- ToySim and Geant4 based Ideal Pattern reco. agree very well
  - Gives solid baseline of what we can achieve
- Many very usefull components already exist
  - Framework
  - Hough tracking
  - GenFit2 interface for track fitting

The Upside



- ToySim and Geant4 based Ideal Pattern reco. agree very well
  - Gives solid baseline of what we can achieve
- Many very usefull components already exist
  - Framework
  - Hough tracking
  - GenFit2 interface for track fitting
    - Kalman fit, DAF etc

Current HEP industry standard

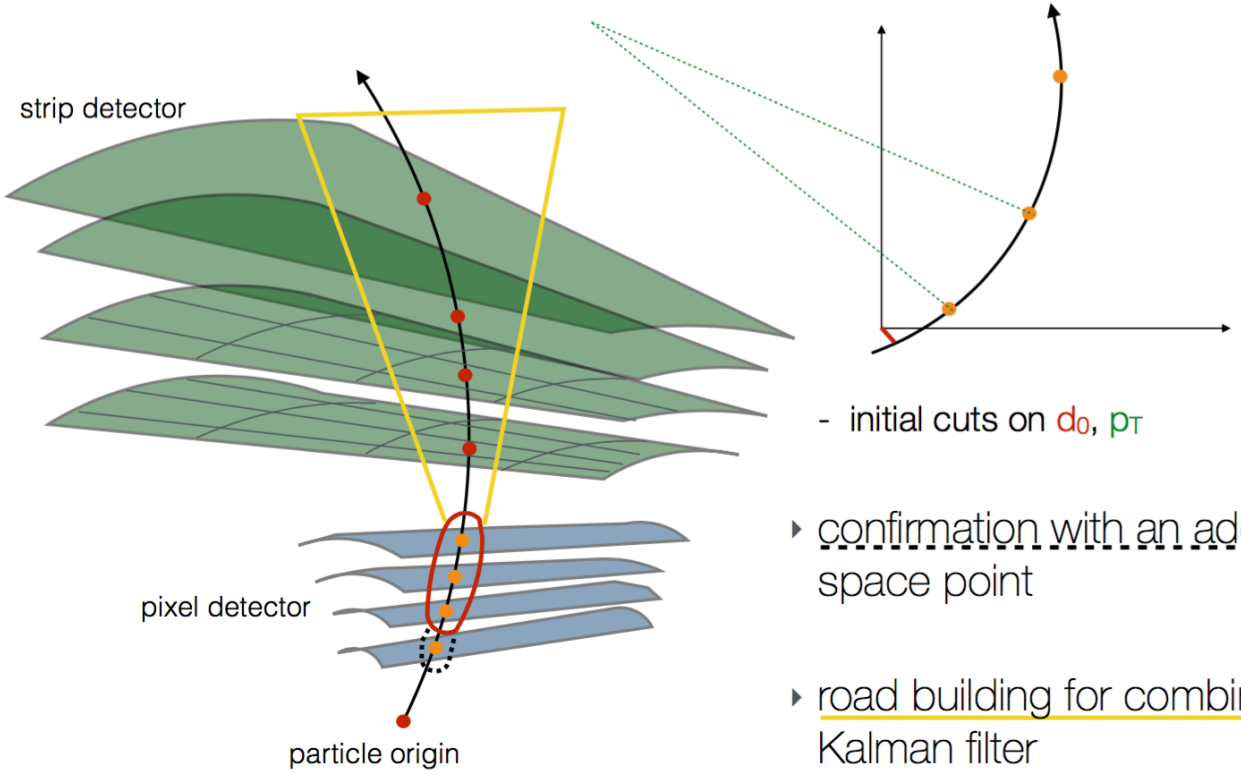
- Kalman filter based combinatorial tracking

The nutshell view - track finding (1)

1)Track Seeding
HEP: inside - out

Space Point seeding

- building triplets of space points



- initial cuts on d_0, p_T

- confirmation with an additional space point

- road building for combinatorial Kalman filter

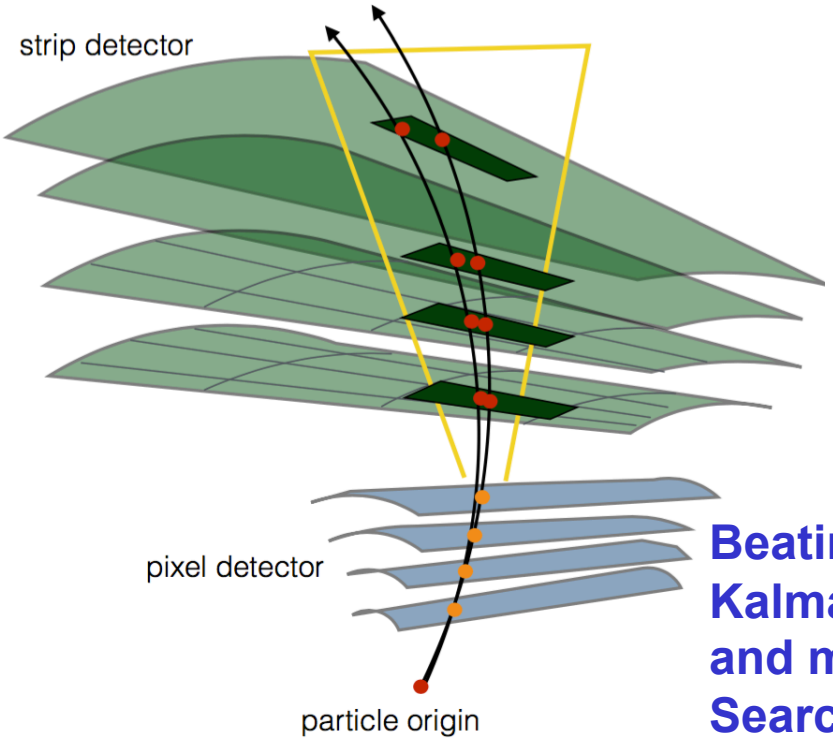
2)Trajectory building

From A. Salzburger, Connecting the Dots 2016

Current HEP industry standard

- Kalman filter based combinatorial tracking

Space point seeded track finding



3) Pick up hits in the search window defined by track seed

If multiple compatible hits are found, clone track and continue propagating both.
One seed -> many trajectories

4) Monitor trajectory quality
After n layers cross without finding a compatible hit -> drop the trajectory

Beating combinatorics:
Kalman filter updates momentum estimate and measurment error with each added point
Search window in subsequent layer shrinks with each added point
In CMS: after 6-7 hits on track the trajectory is defined well enough to find ~1 hit layer after that

From A. Salzburger, Connecting the Dots 2016

Short term ToDo list

- A beautiful standalone implementation of a Kalman filter based combinatorial track finder from scratch will take a lot longer than ~2-3 month we have for the initial task

- Start with a 'proof-of-principal' implementation that is good enough to kick off the MVTX proposal, polish and fine tune later

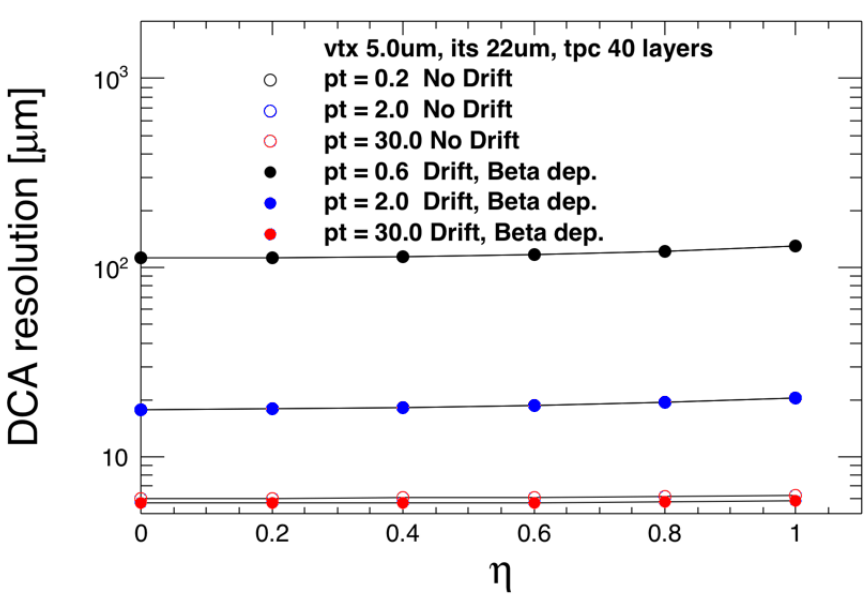
- Replace Hough tracking in the TPC with Kalman propagation
  - Make sure we get the clustering fixed and have good hit positions as input
  - Use Hough tracking in the MAPS (+ ITT ?) to generate track seeds
  - Write track propagation through TPC layers based on a Kalman combinatorial tracking strategy
    - Specifically: add hits to seed trajectories and use GenFit2 Kalman fitter to calculate the Chi2 increment of the hit with respect to the previous trajectory state.
  - Later add generic track seeding with arbitrary layer combinations
    - Outside in vs inside out
    - Iterative tracking steps?
  - Be realistic about the goals
    - Most important goal: Demonstrate we can achieve the nominal parameter resolutions with good reconstruction efficiencies and reasonable fake rates
    - If we can anchor this in full simulations the main part of the studies for the MVTX proposal can be done using some form of fast simulation or parameterized performance.
    - Reuse code we can potentially procure from other experiments (e.g. ALICE)
    - Don't try to win beauty contests for elegantly written code…

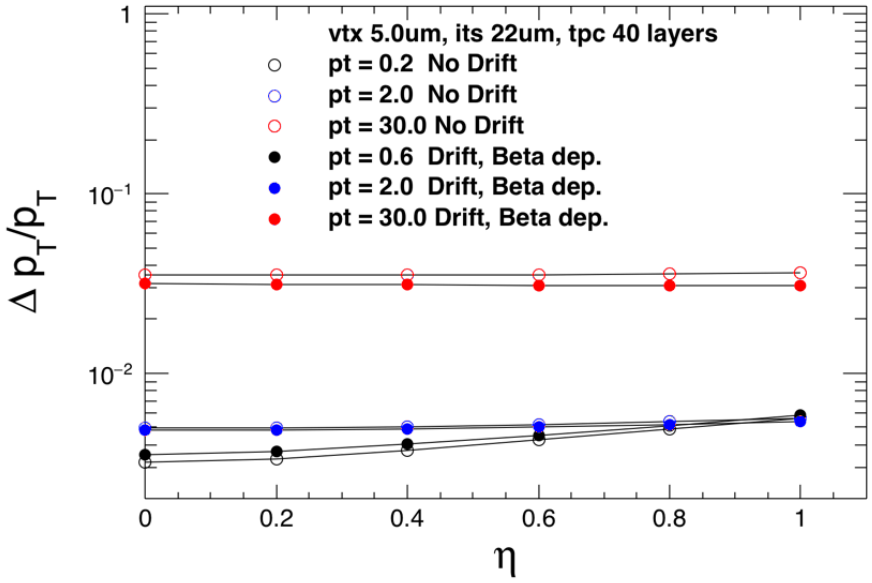
Manpower

- Me obviously…
  - May get some help from UROP students at MIT
- Haiwang expressed interest
- Anybody else?

BACK-UP

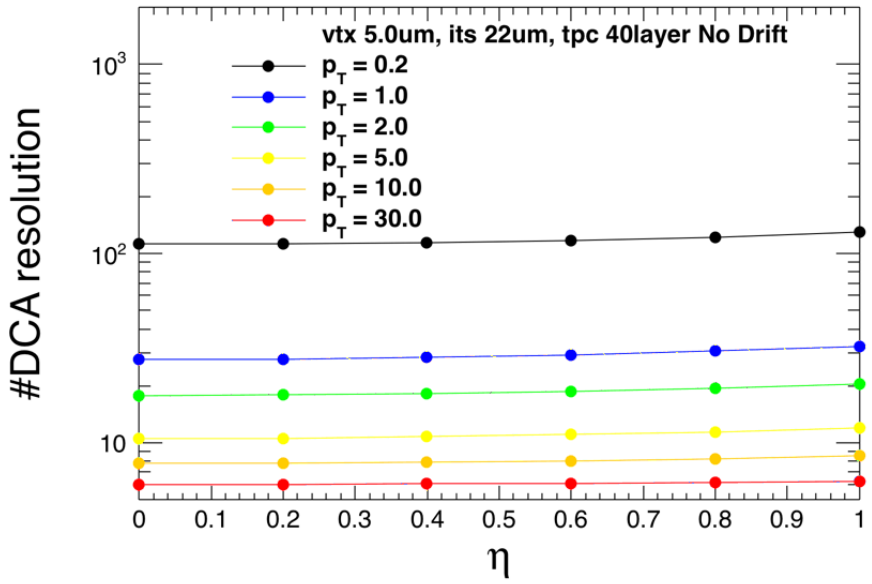
Drift Dependence

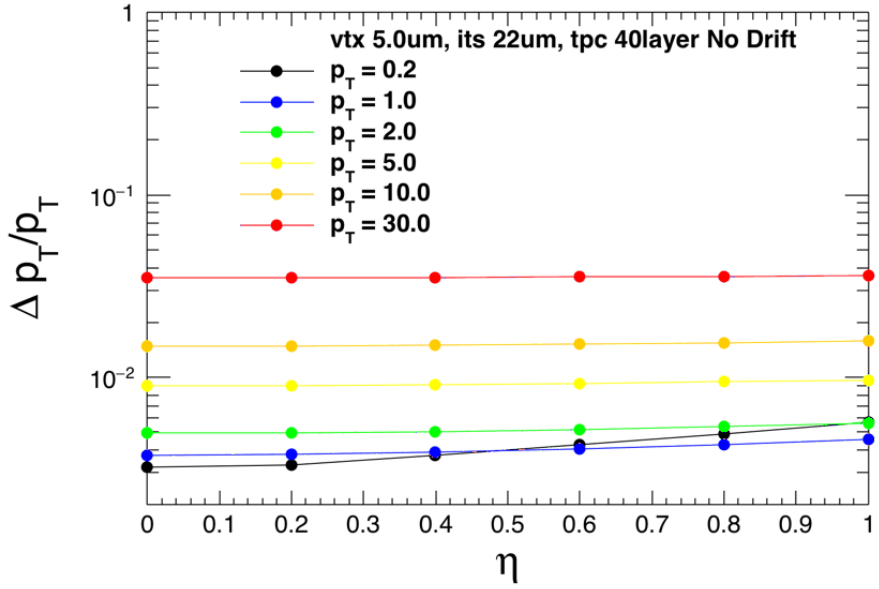


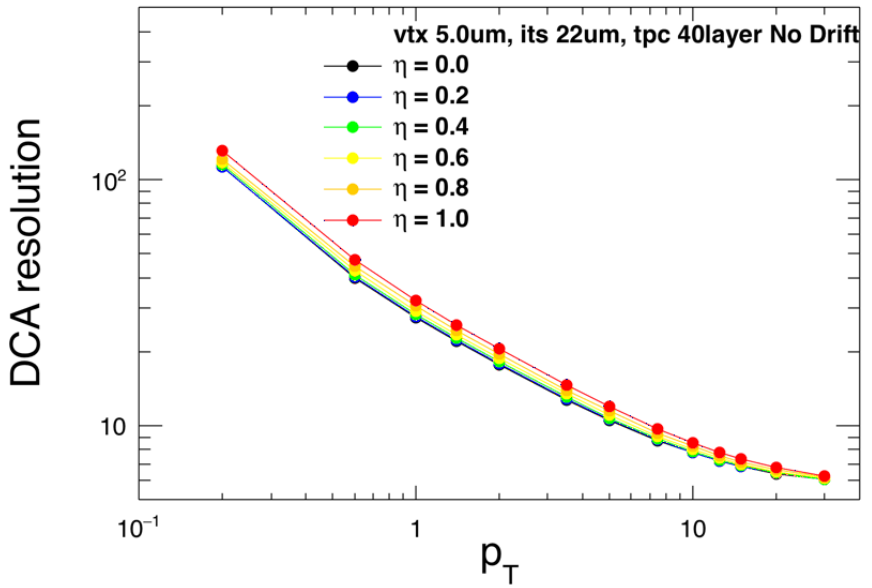


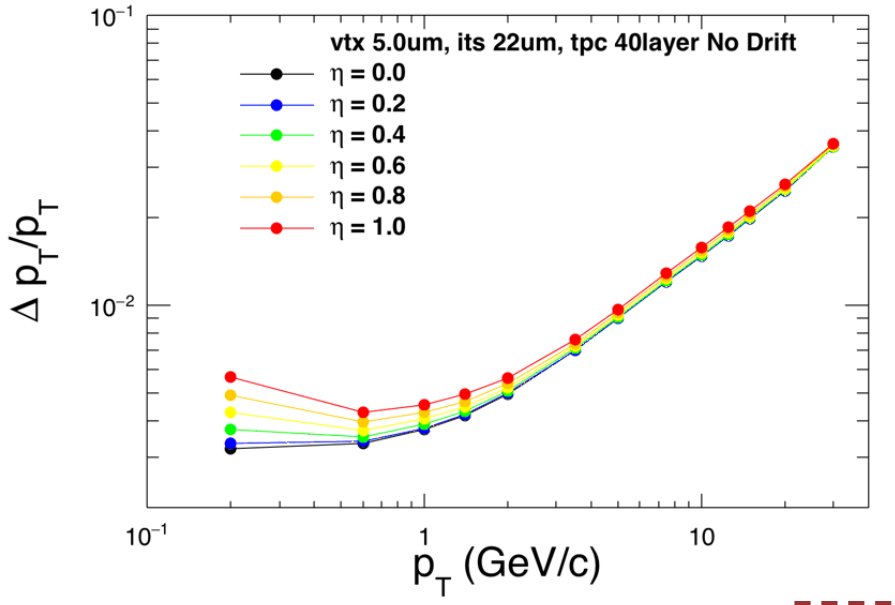
- Visible effect on p_T resolution vs p_T, but nothing dramatic
- Very little change in DCA resolution, as expected

Resolutions without drift dependence

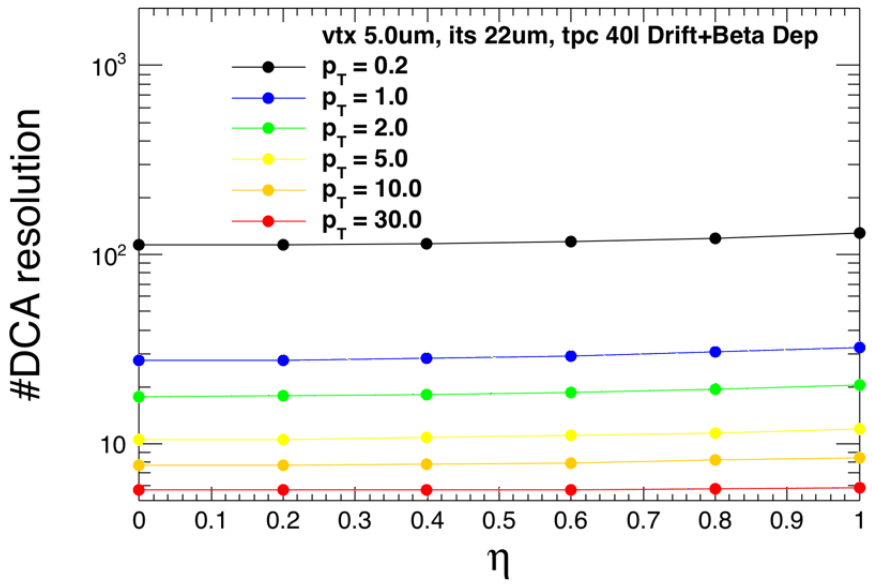


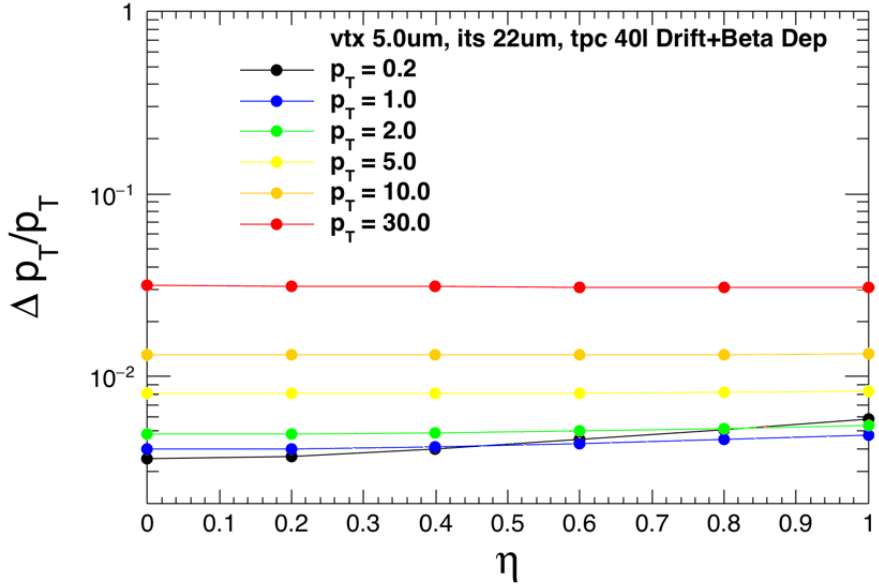


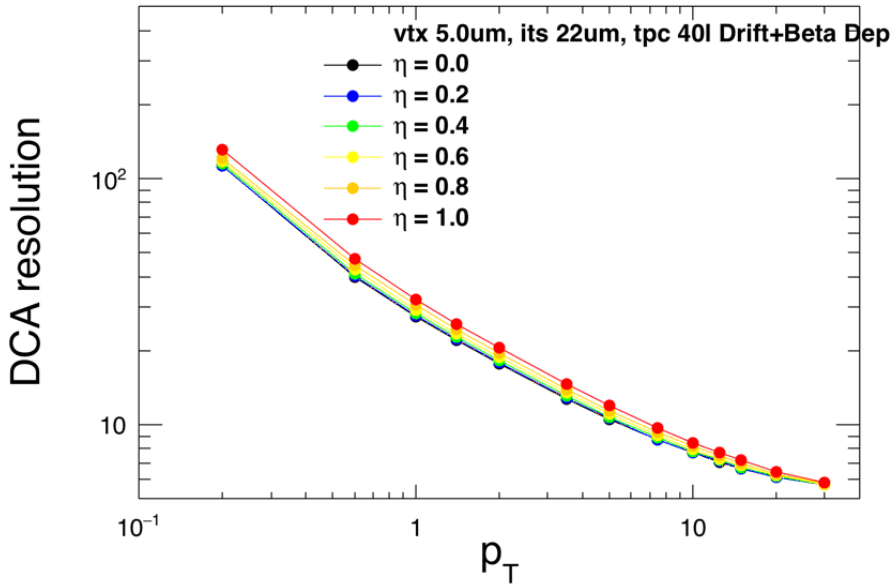


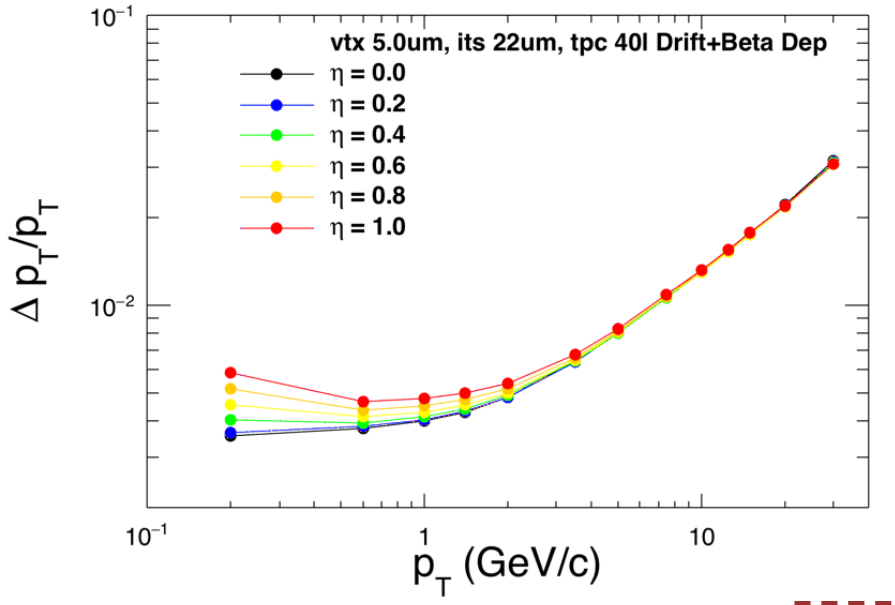


Resolutions, full Drift and Beta dependence

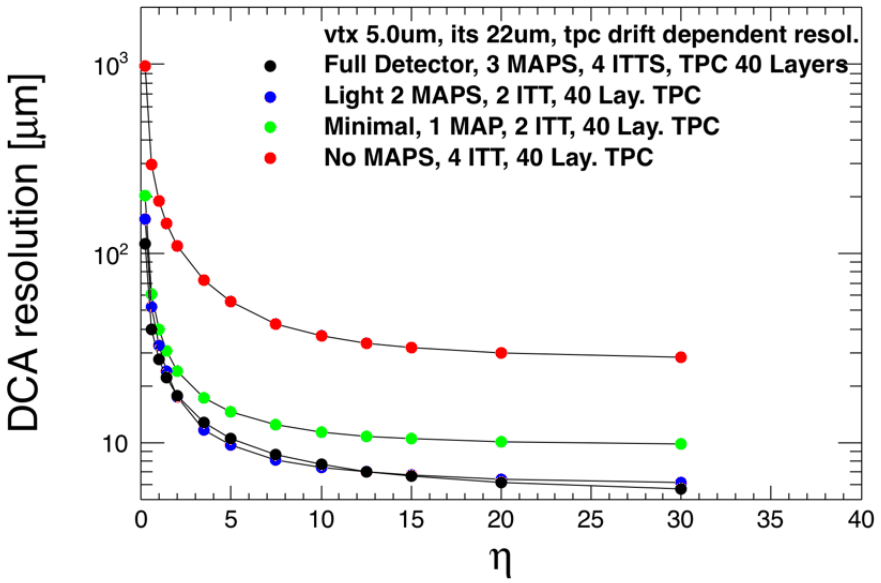


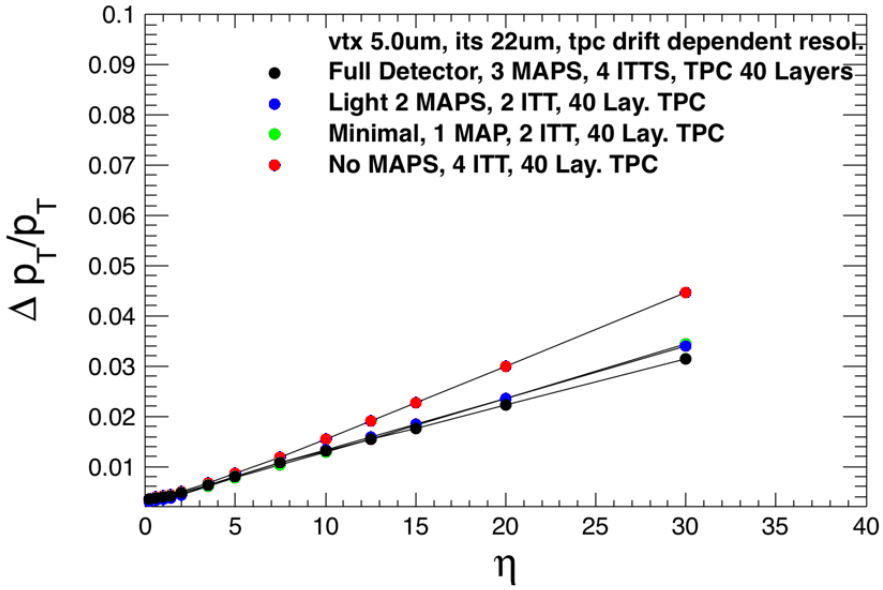






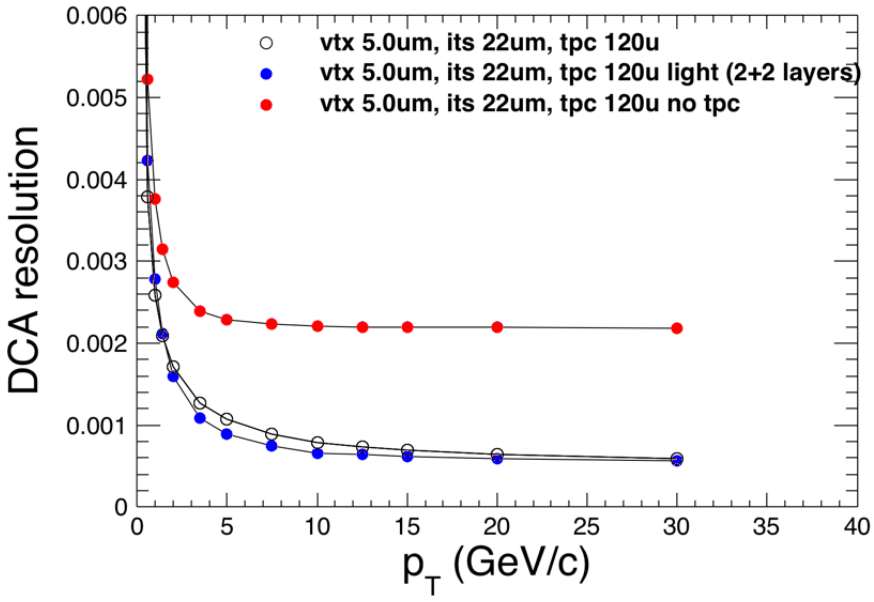
Alternate Detector Configurations

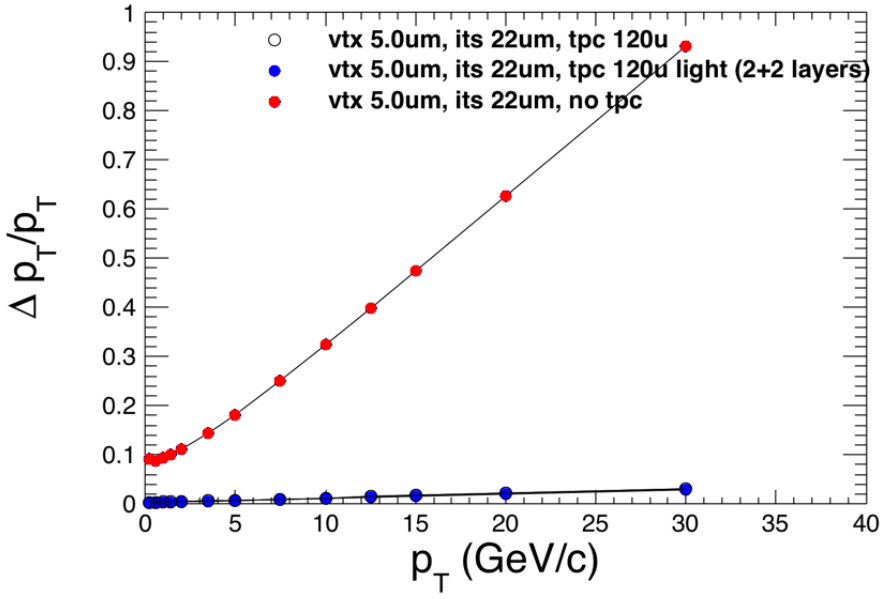




- Alternate detector configurations
  - Default -> 3 MAPS Layers, 4 ITT Layers, 40 Layers TPC
  - Light-> 2 MAPS Layers, 2 ITT Layers to save material budget
    - Slight performance advantage below 10GeV due to lower material budget
  - Minimal -> 1 Maps Layer, 2 ITT Layers, 40 Layers TPC
    - Significant performance decrease (remember 95% hit efficiency per layer)
  - No Maps
    - Likely death sentence for Heavy Flavor program…

Standalone MAPS+ITT performance





MAPS+TPC performance (no ITT)